

*please enter
70.2.0.
8/4/05*

PTO/SB/21 (09-04)

Approved for use through 07/31/2008. OMB 0651-0031

U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

TRANSMITTAL FORM <small>(to be used for all correspondence after initial filing)</small>	Application Number	09766288
	Filing Date	01/03/2001
	First Named Inventor	Dirk Coldewey
	Art Unit	2177
	Examiner Name	Harold E. Dodds, Jr.
Total Number of Pages in This Submission	Attorney Docket Number	

ENCLOSURES (Check all that apply)		
<input type="checkbox"/> Fee Transmittal Form <input type="checkbox"/> Fee Attached <input checked="" type="checkbox"/> Amendment/Reply <input type="checkbox"/> After Final <input type="checkbox"/> Affidavits/declaration(s) <input type="checkbox"/> Extension of Time Request <input type="checkbox"/> Express Abandonment Request <input type="checkbox"/> Information Disclosure Statement <input type="checkbox"/> Certified Copy of Priority Document(s) <input type="checkbox"/> Reply to Missing Parts/ Incomplete Application <input type="checkbox"/> Reply to Missing Parts under 37 CFR 1.52 or 1.53	<input checked="" type="checkbox"/> Drawing(s) <input type="checkbox"/> Licensing-related Papers <input type="checkbox"/> Petition <input type="checkbox"/> Petition to Convert to a Provisional Application <input type="checkbox"/> Power of Attorney, Revocation <input type="checkbox"/> Change of Correspondence Address <input type="checkbox"/> Terminal Disclaimer <input type="checkbox"/> Request for Refund <input type="checkbox"/> CD, Number of CD(s) _____ <input type="checkbox"/> Landscape Table on CD	<input type="checkbox"/> After Allowance Communication to TC <input type="checkbox"/> Appeal Communication to Board of Appeals and Interferences <input type="checkbox"/> Appeal Communication to TC (Appeal Notice, Brief, Reply Brief) <input type="checkbox"/> Proprietary Information <input type="checkbox"/> Status Letter <input type="checkbox"/> Other Enclosure(s) (please identify below):
Remarks Please see separate remarks page between markup copy of application and the non-markup copy of the transmission.		

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT		
Firm Name		
Signature	/Dirk Coldewey/	
Printed name	Dirk Coldewey	
Date	7/27/2005	Reg. No.

CERTIFICATE OF TRANSMISSION/MAILING		
I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below:		
Signature	/Dirk Coldewey/	
Typed or printed name	Dirk Coldewey	Date 7/27/2005

This collection of information is required by 37 CFR 1.5. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.11 and 1.14. This collection is estimated to 2 hours to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

If you need assistance in completing the form, call 1-800-PTO-9199 and select option 2.

(Replacement Sheet — drawings separated from claims section)

```

Traverse( forest_ptr forest )
{
    /* local variables */
    stack stacks[PipeDepth]; /* PipeDepth stacks */
    tree_ptr n;
    int i, trees_left = PipeDepth;
    struct {
        tree_ptr node;
        stack_ptr stack;
    } traversal[PipeDepth]; /* traversal state descriptor */

    /* prologue */
    for ( i=0; i<PipeDepth; i++ ) {
        traversal[i].node = forest->root[i];
        traversal[i].stack = &stack[i];
        PREFETCH(forest->root[i], sizeof(forest->root[i]));
    }

    /* steady state */
    while ( trees_left ) {
        for ( i=0; i<trees_left; i++ ) {
            if ( traversal[i].node->left ) {
                traversal[i].stack->push( traversal[i].node->left );
                traversal[i].node = traversal[i].node->left;
            } else {
                n = traversal[i].stack->pop();
                if ( n == NULL ) { /* done with tree i */
                    trees_left--;
                    if ( i != trees_left )
                        SWAP( &traversal[i], &traversal[trees_left] );
                }
                process( n );
                traversal[i].node = n->right;
            }
            PREFETCH( traversal[i].node );
        }
    }
}

```

Figure 7: Example of a Pipelined Tree Traversal

(Replacement Sheet — drawings separated from claims)

```

Traverse( tree_ptr tree )
{
    /* local variables */
    . . .

    /* level-order traversal prologue */
    PREFETCH( tree->root );
    enqueue( src_queue, tree->root );
    for ( i=0, accumulating=true; accumulating; i++ ) {
        n = dequeue( src_queue );
        if ( n == NULL )
            return; /* we're done */
        process( n->data );

        if ( n->left != NULL ) {
            PREFETCH( n->left );
            enqueue( dst_queue, n->left );
        }
        if ( n->right != NULL ) {
            PREFETCH( n->right );
            enqueue( dst_queue, n->right );
        }
        if ( src_queue->size + dst_queue->size < PipeDepth ) {
            if ( i >= src_queue->size )
                SWAP( src_queue, dst_queue );
        } else {
            accumulating = false;
            while ( src_queue->size > 0 ) {
                traversal[trees_left].node = dequeue( src_queue );
                traversal[trees_left].stack = stack[trees_left];
                trees_left++;
            }
            while ( dst_queue->size > 0 ) {
                traversal[trees_left].node = dequeue( dst_queue );
                traversal[trees_left].stack = stack[trees_left];
                trees_left++;
            }
        }
    }

    /* steady state loop */
    . . .
}

```

Figure 8: Example of a pipelined level-order tree traversal.